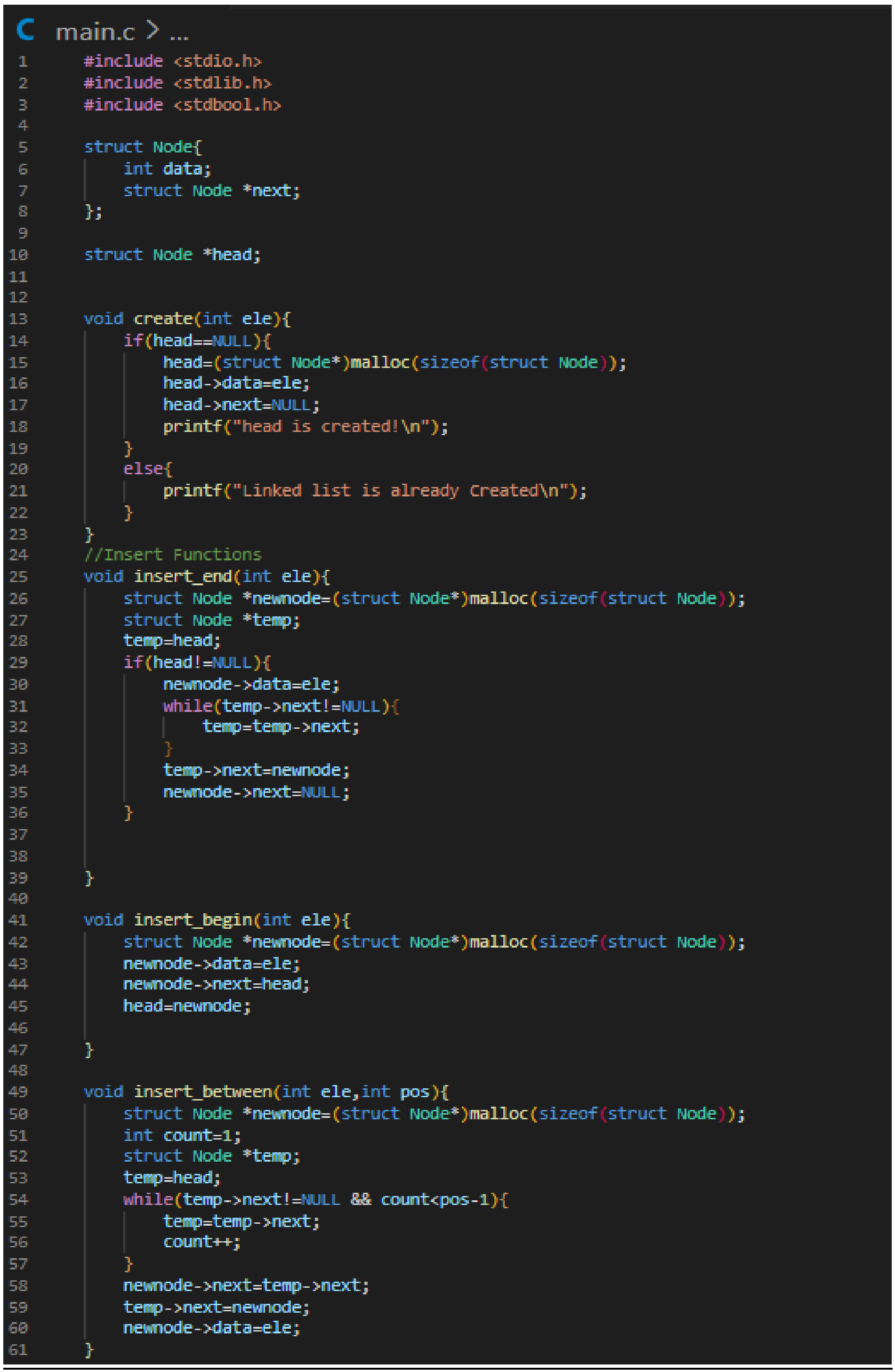
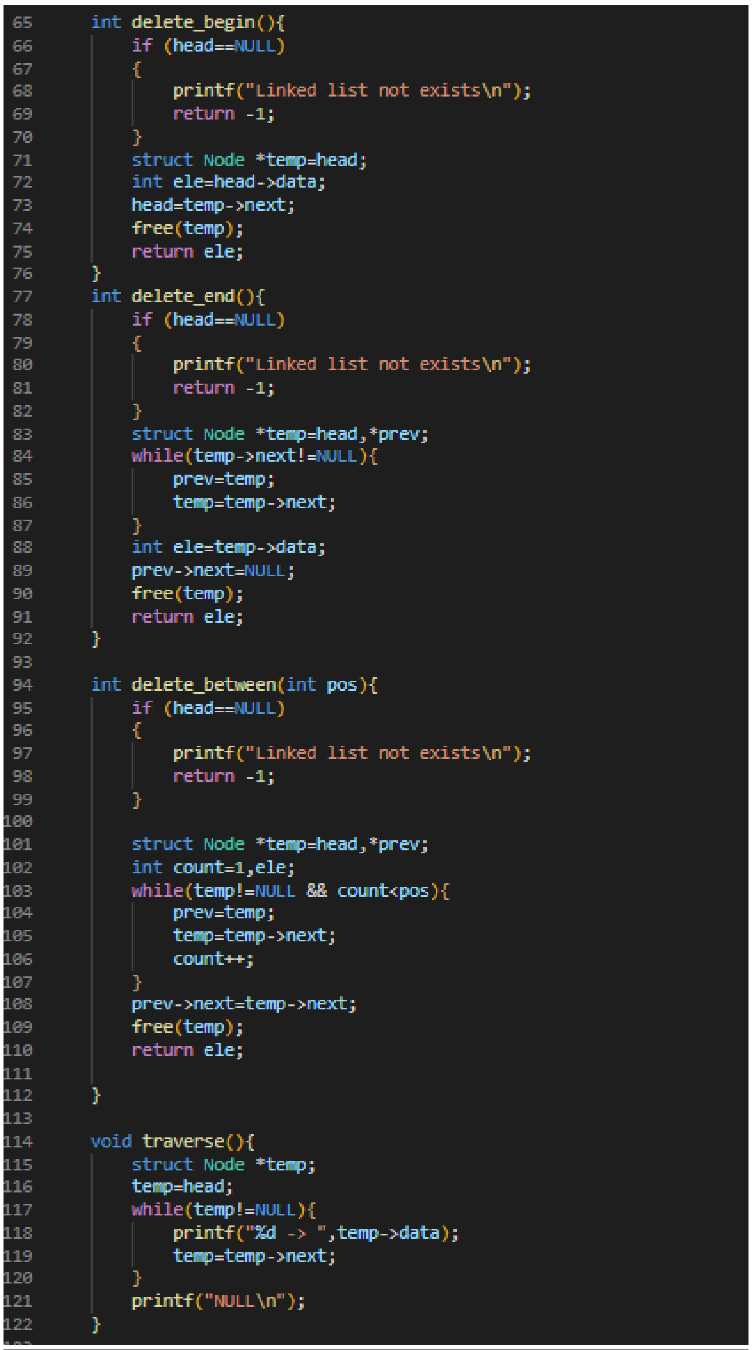
PRACTICAL-4

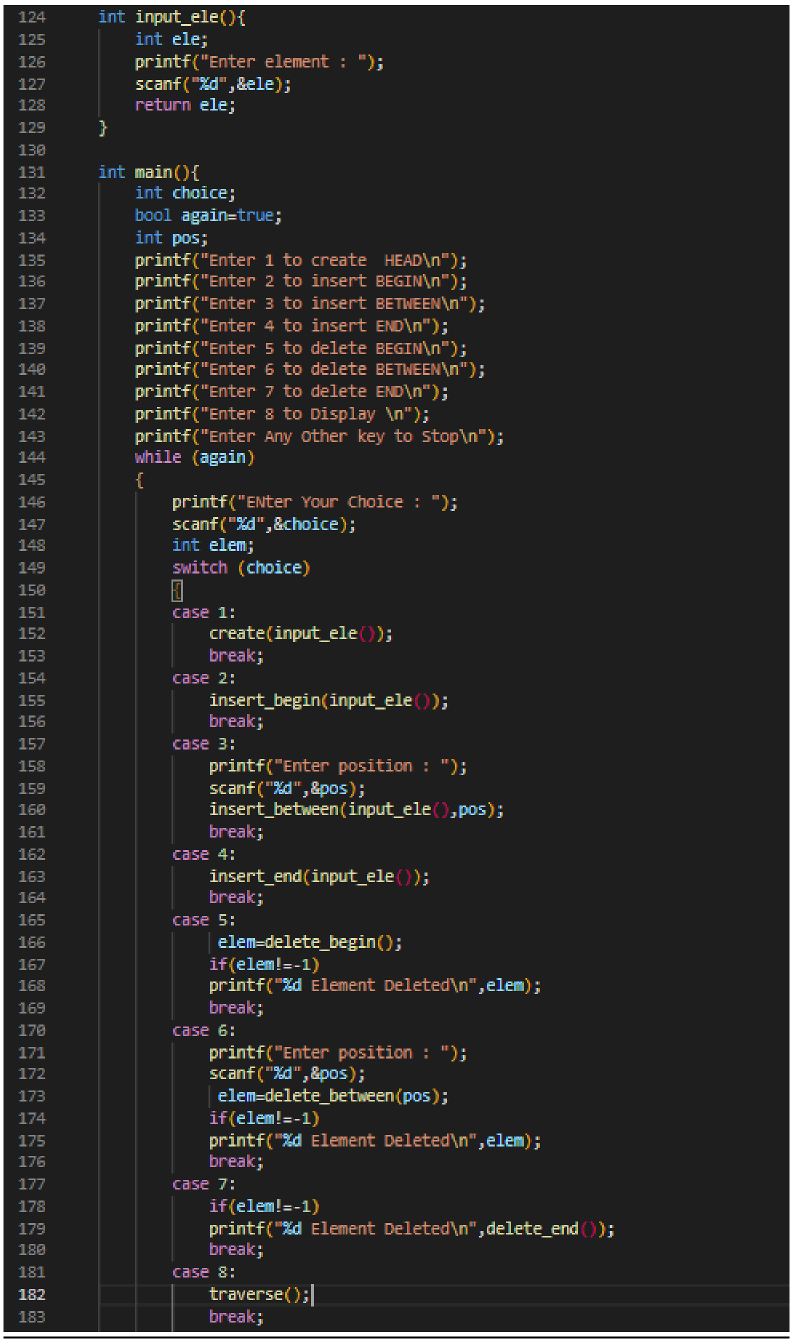
CODE:



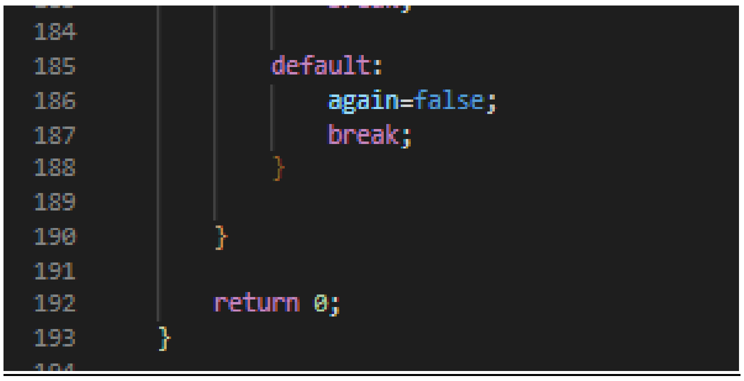
main.c > ...

```c
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>

struct Node{
    int data;
    struct Node *next;
};

struct Node *head;


void create(int ele){
    if(head==NULL){
        head=(struct Node*)malloc(sizeof(struct Node));
        head->data=ele;
        head->next=NULL;
        printf("head is created!\n");
    }
    else{
        printf("Linked list is already Created\n");
    }
}
//Insert Functions
void insert_end(int ele){
    struct Node *newnode=(struct Node*)malloc(sizeof(struct Node));
    struct Node *temp;
    temp=head;
    if(head!=NULL){
        newnode->data=ele;
        while(temp->next!=NULL){
            temp=temp->next;
        }
        temp->next=newnode;
        newnode->next=NULL;
    }


}

void insert_begin(int ele){
    struct Node *newnode=(struct Node*)malloc(sizeof(struct Node));
    newnode->data=ele;
    newnode->next=head;
    head=newnode;

}

void insert_between(int ele,int pos){
    struct Node *newnode=(struct Node*)malloc(sizeof(struct Node));
    int count=1;
    struct Node *temp;
    temp=head;
    while(temp->next!=NULL && count<pos-1){
        temp=temp->next;
        count++;
    }
    newnode->next=temp->next;
    temp->next=newnode;
    newnode->data=ele;
}
```


```c
int delete_begin(){
    if (head==NULL)
    {
        printf("Linked list not exists\n");
        return -1;
    }
    struct Node *temp=head;
    int ele=head->data;
    head=temp->next;
    free(temp);
    return ele;
}
int delete_end(){
    if (head==NULL)
    {
        printf("Linked list not exists\n");
        return -1;
    }
    struct Node *temp=head,*prev;
    while(temp->next!=NULL){
        prev=temp;
        temp=temp->next;
    }
    int ele=temp->data;
    prev->next=NULL;
    free(temp);
    return ele;
}

int delete_between(int pos){
    if (head==NULL)
    {
        printf("Linked list not exists\n");
        return -1;
    }

    struct Node *temp=head,*prev;
    int count=1,ele;
    while(temp!=NULL && count<pos){
        prev=temp;
        temp=temp->next;
        count++;
    }
    prev->next=temp->next;
    free(temp);
    return ele;

}

void traverse(){
    struct Node *temp;
    temp=head;
    while(temp!=NULL){
        printf("%d -> ",temp->data);
        temp=temp->next;
    }
    printf("NULL\n");
}
```



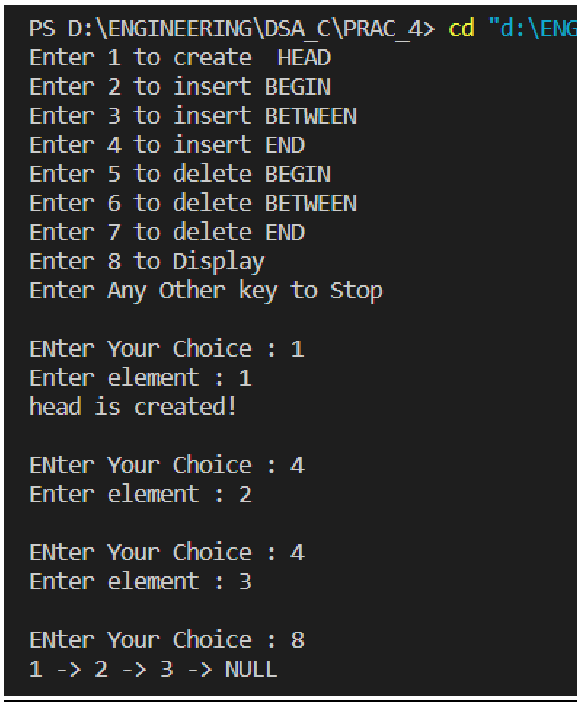
```c
int input_ele(){
    int ele;
    printf("Enter element : ");
    scanf("%d",&ele);
    return ele;
}

int main(){
    int choice;
    bool again=true;
    int pos;
    printf("Enter 1 to create  HEAD\n");
    printf("Enter 2 to insert BEGIN\n");
    printf("Enter 3 to insert BETWEEN\n");
    printf("Enter 4 to insert END\n");
    printf("Enter 5 to delete BEGIN\n");
    printf("Enter 6 to delete BETWEEN\n");
    printf("Enter 7 to delete END\n");
    printf("Enter 8 to Display \n");
    printf("Enter Any Other key to Stop\n");
    while (again)
    {
        printf("ENter Your Choice : ");
        scanf("%d",&choice);
        int elem;
        switch (choice)
        {
        case 1:
            create(input_ele());
            break;
        case 2:
            insert_begin(input_ele());
            break;
        case 3:
            printf("Enter position : ");
            scanf("%d",&pos);
            insert_between(input_ele(),pos);
            break;
        case 4:
            insert_end(input_ele());
            break;
        case 5:
             elem=delete_begin();
            if(elem!=-1)
            printf("%d Element Deleted\n",elem);
            break;
        case 6:
            printf("Enter position : ");
            scanf("%d",&pos);
             elem=delete_between(pos);
            if(elem!=-1)
            printf("%d Element Deleted\n",elem);
            break;
        case 7:
            if(elem!=-1)
            printf("%d Element Deleted\n",delete_end());
            break;
        case 8:
            traverse();
            break;
```



```c

        default:
            again=false;
            break;
        }

    }

    return 0;
}
```

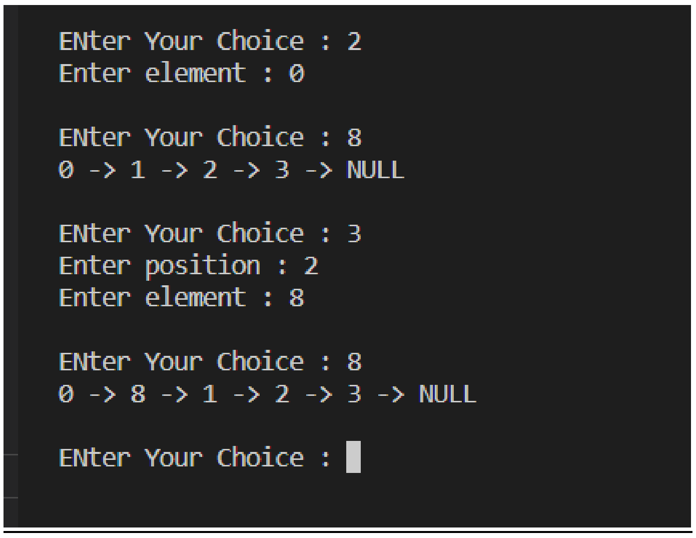
Output:

1)Creating Head and append:



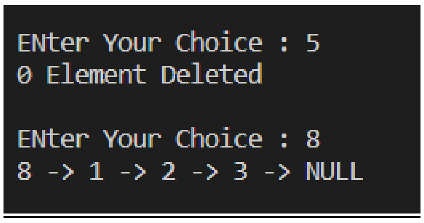
```
PS D:\ENGINEERING\DSA_C\PRAC_4> cd "d:\ENG
Enter 1 to create  HEAD
Enter 2 to insert BEGIN
Enter 3 to insert BETWEEN
Enter 4 to insert END
Enter 5 to delete BEGIN
Enter 6 to delete BETWEEN
Enter 7 to delete END
Enter 8 to Display
Enter Any Other key to Stop

ENter Your Choice : 1
Enter element : 1
head is created!

ENter Your Choice : 4
Enter element : 2

ENter Your Choice : 4
Enter element : 3

ENter Your Choice : 8
1 -> 2 -> 3 -> NULL
```

2)Insert Begin and in between:



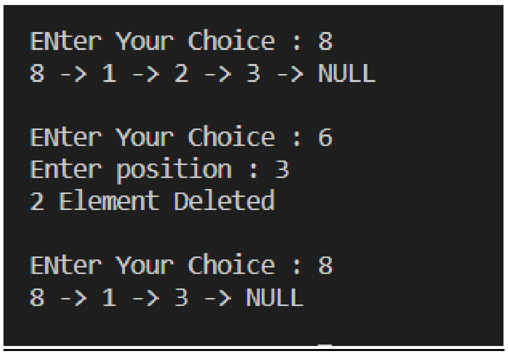
```
ENter Your Choice : 2
Enter element : 0

ENter Your Choice : 8
0 -> 1 -> 2 -> 3 -> NULL

ENter Your Choice : 3
Enter position : 2
Enter element : 8

ENter Your Choice : 8
0 -> 8 -> 1 -> 2 -> 3 -> NULL

ENter Your Choice :
```

3)Delete Begin:



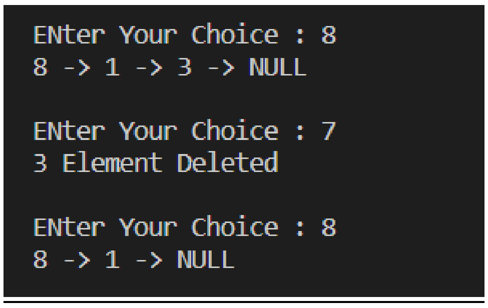
```
ENter Your Choice : 5
0 Element Deleted

ENter Your Choice : 8
8 -> 1 -> 2 -> 3 -> NULL
```

4)Delete Between:



```
ENter Your Choice : 8
8 -> 1 -> 2 -> 3 -> NULL

ENter Your Choice : 6
Enter position : 3
2 Element Deleted

ENter Your Choice : 8
8 -> 1 -> 3 -> NULL
```

5)Delete End:



```
ENter Your Choice : 8
8 -> 1 -> 3 -> NULL

ENter Your Choice : 7
3 Element Deleted

ENter Your Choice : 8
8 -> 1 -> NULL
```